Mitglied der Helmholtz-Gemeinschaft

JÜLICH FORSCHUNGSZENTRUM

Green's function electrostatics solver with boundary conditions

July 23, 2013 | Benedikt Steinbusch

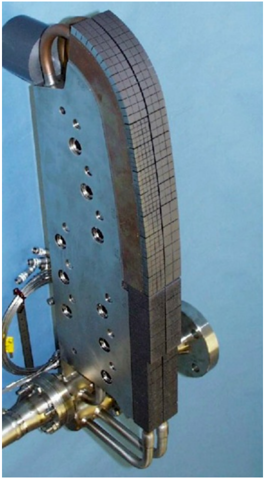
Introduction

Context

- plasma wall interaction
- complex geometry: castellated tiles
- metal walls, free charges

Agenda

- Solving electrostatics: different methods
- Tree code: algorithm details
- Boundary conditions



M. Hellwig (IEK-4)

Solving electrostatics

with and without grids

Poisson equation

$$\Delta\,\phi(x) = \rho(x) \quad x \in \Omega$$

Solving electrostatics

with and without grids

Poisson equation

$$\Delta\,\phi(x) = \rho(x) \quad x \in \Omega$$

Grid based method

- discretise space
  - finite differences

    $$x \in \mathbb{R} \to \Delta x\; i \in \Delta x\; \mathbb{Z}$$

  - finite elements

    $$\Omega = \cup_i\; \Delta\Omega_i$$

- system of linear equations
- approximation of ϕ known on whole Ω

Solving electrostatics

with and without grids

Poisson equation

$$\Delta\, \phi(x) = \rho(x) \quad x \in \Omega$$

Grid based method

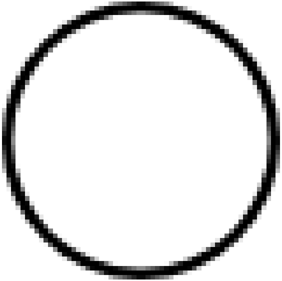
- discretise space
  - finite differences

    $$x \in \mathbb{R} \to \Delta x\; i \in \Delta x\; \mathbb{Z}$$

  - finite elements

    $$\Omega = \cup_i\; \Delta\Omega_i$$

- system of linear equations
- approximation of ϕ known on whole Ω



Solving electrostatics

with and without grids

Poisson equation

$$\Delta\,\phi(x) = \rho(x) \quad x \in \Omega$$

Green's function based method

- use discrete source distribution

$$\rho(x) = \sum_j q_j \delta(x - x_j)$$

- and Green's function

$$\phi(x) = \int_\Omega \rho(x')G(x, x')\,\mathrm{d}V'$$

- ϕ known at arbitrary x

$$\phi(x) = \sum_j q_j G(x, x_j)$$

Solving electrostatics

with and without grids

Poisson equation

$$\Delta\,\phi(x) = \rho(x) \quad x \in \Omega$$

Green's function based method

- use discrete source distribution

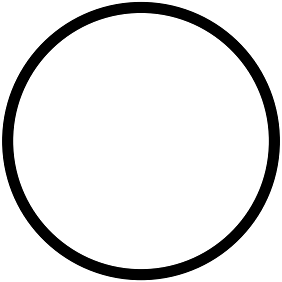
$$\rho(x) = \sum_j q_j \delta(x - x_j)$$

- and Green's function

$$\phi(x) = \int_\Omega \rho(x') G(x, x')\,\mathrm{d}V'$$

- ϕ known at arbitrary x

$$\phi(x) = \sum_j q_j G(x, x_j)$$



Speeding things up

multipole expansion of Green's function

- for particle dynamics, need ϕ at every x_i:

$$\phi(x_i) = \sum_{j=1}^{N} q_j G(x_i, x_j) \to \phi_i = G_{ij} q_j \quad i = 1 \dots N$$

- G is dense (long range interaction)
- matrix-vector product costs $\mathcal{O}(N^2)$ operations
- group distant interaction partners:

$$G(x, x') \approx \sum_{k=0}^{p} \chi_k(x, \bar{x}) \psi_k(x', \bar{x}) \qquad Q_{mk} = \sum_{j \in m} q_j \psi_k \left(x_j, \bar{x}_m\right)$$

$$\phi(x_i) \approx= \sum_{m \in M_i} \sum_{k=0}^{p} \chi_k(x_i, \bar{x}_m) Q_{mk} \qquad i = 1 \dots N$$

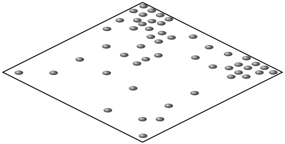
- for the tree code method $|M_i| \approx \log(N) \to \mathcal{O}(N \log(N))$ operations

Keeping one's distance

stratified multipole moments, organized as a tree

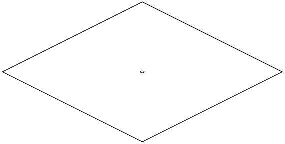
[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

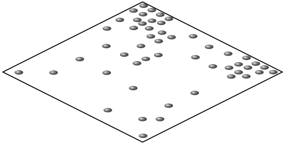
- starting from a cloud of particles



Keeping one's distance

stratified multipole moments, organized as a tree



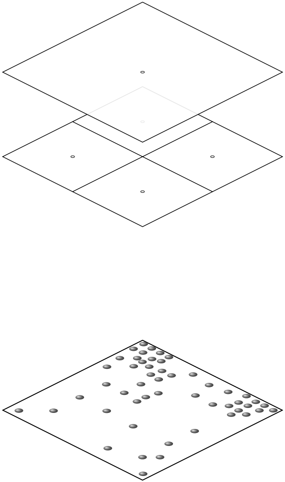


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles

Keeping one's distance

stratified multipole moments, organized as a tree

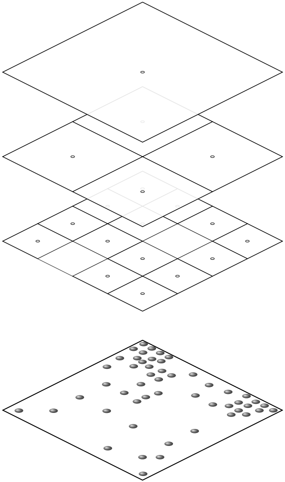


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .

Keeping one's distance

stratified multipole moments, organized as a tree

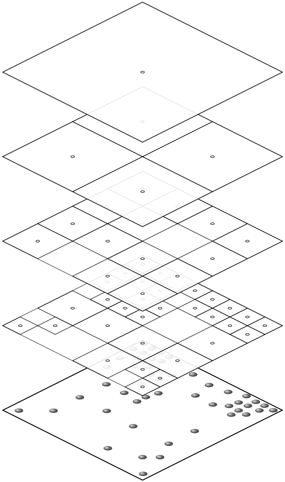


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until…

Keeping one's distance

stratified multipole moments, organized as a tree

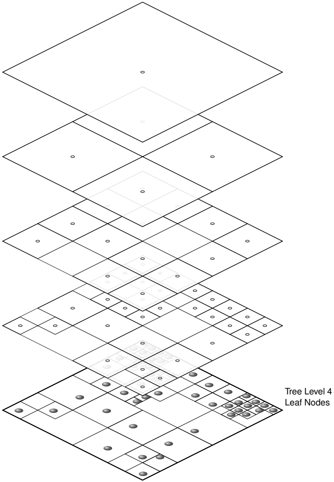


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .

Keeping one’s distance

stratified multipole moments, organized as a tree

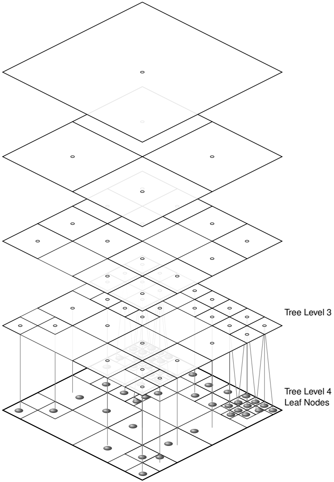


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .
- . . . every particle lies in its own box

Keeping one’s distance

stratified multipole moments, organized as a tree

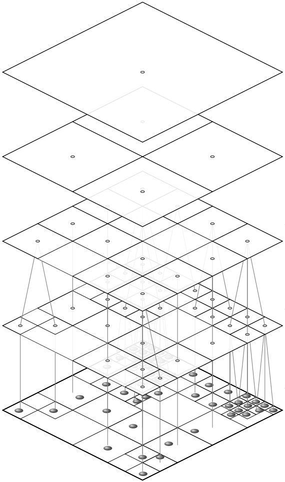


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .
- . . . every particle lies in its own box
- calculate moments on lowest level and propagate upwards

Keeping one’s distance

stratified multipole moments, organized as a tree

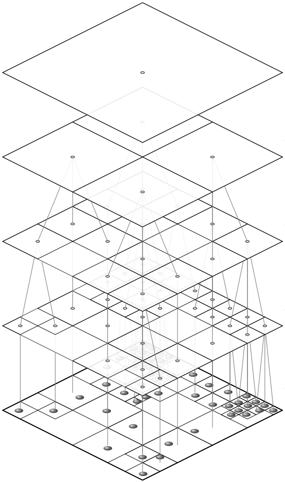


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .
- . . . every particle lies in its own box
- calculate moments on lowest level and propagate upwards

Keeping one's distance

stratified multipole moments, organized as a tree

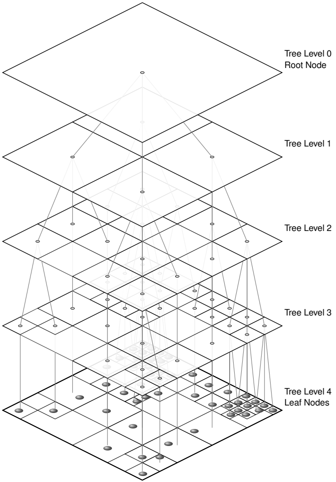


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until. . .
- . . . every particle lies in its own box
- calculate moments on lowest level and propagate upwards

Keeping one's distance

stratified multipole moments, organized as a tree

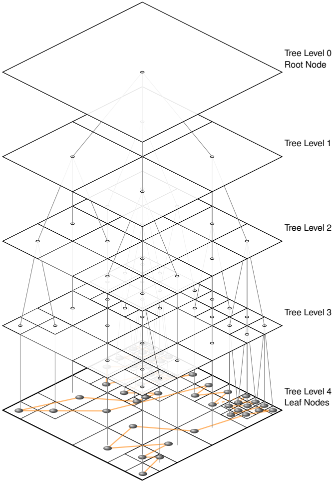


[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

- starting from a cloud of particles
- construct a box containing all particles
- recursively subdivide until...
- ...every particle lies in its own box
- calculate moments on lowest level and propagate upwards

Distributed (sub-)trees

space filling curves and branches



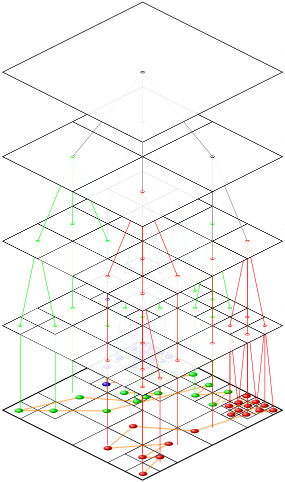
[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

[M. S. Warren & J. K. Salmon, Proc. of the 1993 ACM/IEEE Conf. on Supercomputing, 12–21 (1993)]

- sort particles along space filling curve
- partition according to “workload” during last timestep

Distributed (sub-)trees

space filling curves and branches



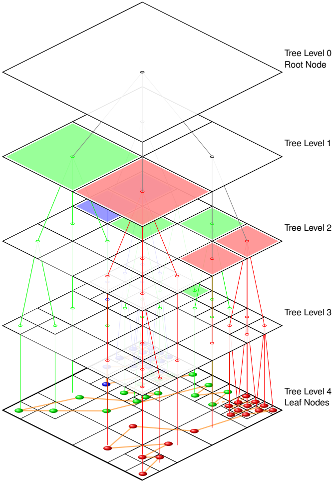
[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

[M. S. Warren & J. K. Salmon, Proc. of the 1993 ACM/IEEE Conf. on Supercomputing, 12–21 (1993)]

- sort particles along space filling curve
- partition according to “workload” during last timestep
- build a local tree on each rank

Distributed (sub-)trees

space filling curves and branches



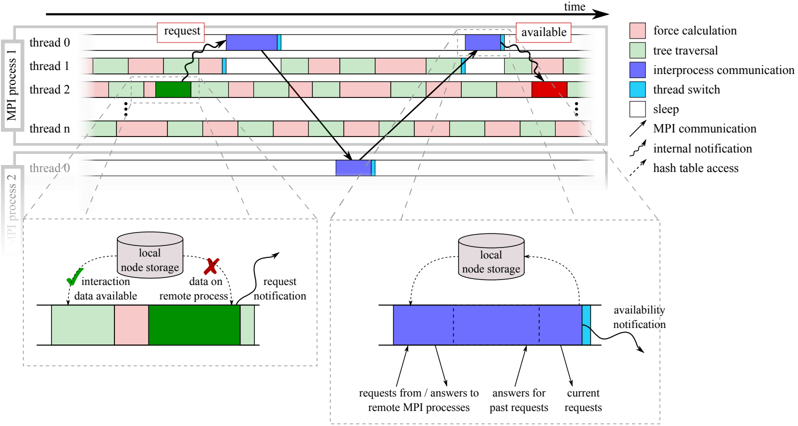
[J. Barnes & P. Hut, Nature **324**, 446-449 (1986)]

[M. S. Warren & J. K. Salmon, Proc. of the 1993 ACM/IEEE Conf. on Supercomputing, 12–21 (1993)]

- sort particles along space filling curve
- partition according to “workload” during last timestep
- build a local tree on each rank
- identify and exchange “branch” nodes
- update tree between branches and root node

Communication among trees

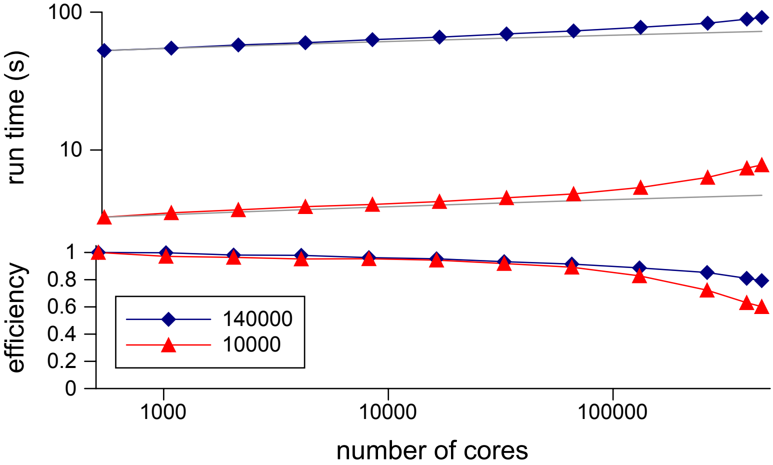
and hybrid parallelization



[M. Winkel *et al.*, Comp. Phys. Comm. 187, 880–889 (2012)]

Big trees

recent results on JUQUEEN



No walls in space

boundary conditions

Boundary value problem

$$\Delta\,\phi(x) = \rho(x) \quad x \in \Omega$$

$$\phi(x) = \bar{\phi}(x) \quad x \in \partial\Omega_\mathrm{D} \qquad\qquad \partial_\mathbf{n}\phi(x) = \bar{q}(x) \quad x \in \partial\Omega_\mathrm{N}$$

- boundary conditions are essential (well-posedness)
- natural with grid based approach
  - e.g. difference operator at grid's edge
- implicit assumption in Green's function approach previously shown:

$$\partial\Omega \to \partial\mathbb{R}^n \quad \bar{\phi} = \bar{q} = 0$$

  - fulfilled by "free space" Green's function
  - astrophysics heritage of the tree code method

The complete picture

Representation integral

$$\phi(x) = \int_{\Omega} \rho(x')G(x,x')\,\mathrm{d}V' + \int_{\partial\Omega} \left[\bar{\phi}(x')\partial'_{\mathbf{n}}G(x,x') - \bar{q}(x')G(x,x')\right]\,\mathrm{d}S'$$

- ρ represents discrete sources
- $\bar{\phi}$ and $\bar{q}$ in general are continuous
- discretise the boundary into elements $\partial\Omega_k$:

$$\phi(x_i) = \sum_j q_j G(x_i,x_j) + \sum_k \int_{\partial\Omega_k} \left[\bar{\phi}(x')\partial'_{\mathbf{n}}G(x_i,x') - \bar{q}(x')G(x_i,x')\right]\,\mathrm{d}S'$$

$$\phi_i = G_{ij}q_j + F_{ik}\cdot\bar{\phi}_k - G_{ik}\cdot\bar{q}_k$$

- can be accelerated with tree code method
- lowers dimension of discretisation by one

Solving the boundary value problem

with the Boundary Element Method

- normally $\bar{\phi}$ given on $\partial\Omega_\mathrm{D}$, $\bar{q}$ given on $\partial\Omega_\mathrm{N}$, with $\partial\Omega_\mathrm{D} \cup \partial\Omega_\mathrm{N} = \partial\Omega$
  e.g. Dirichlet boundary conditions $\bar{\phi}$ fixed on whole $\partial\Omega$, $\bar{q}$ unknown
- construct a system of equations by collocation
- for every discretisation element $\partial\Omega_i$ of $\partial\Omega$ evaluate

$$\lim_{x \to x_i} \phi(x) = \ldots = G_{ij} q_j + F_{ik} \cdot \bar{\phi}_k - G_{ik} \cdot \bar{q}_k$$

  where x_i is e.g. the center of $\partial\Omega_i$
- rearrange:

$$G_{ik} \cdot \bar{q}_k = \hat{F}_{ik} \cdot \bar{\phi}_k + G_{ij} q_j$$
$$\hat{F}_{ik} = F_{ik} - \delta_{ik}$$

- linear system $A\ x = b$ can be solved iteratively, tree code method accelerates:
  - calculation of RHS vector b
  - multiplication of system matrix A with solution vector x
- rearranging A and b yields: mixed Dirichlet and Neumann, periodic, metal wall at floating potential

Implementation

- Green's function tree code method provided by `pepc`, developed at JSC
- Iterative solver provided by `PETSc`, developed at ANL, ...
- Both MPI parallel, `PETSc` widely available
- 2D version with straight, constant boundary elements
- Mixed boundary conditions

Implementation

- Green's function tree code method provided by `pepc`, developed at JSC
- Iterative solver provided by `PETSc`, developed at ANL, ...
- Both MPI parallel, `PETSc` widely available
- 2D version with straight, constant boundary elements
- Mixed boundary conditions

Algorithm

For every timestep:

1. From plasma particles q_i and prescribed boundary conditions $\bar{\phi}_i$ and $\bar{q}_i$ calculate RHS vector b using tree code
2. Iterative solver provides solution vector x, calculate interactions $A\,x$
3. Repeat step 2 until $A\,x$ converges to b
4. Using solution x and given quantities, calculate forces on particles
5. Advance particles by integrating equation of motion
6. Go to 1

Thank you!

More boundary conditions

Mixed Dirichlet and Neumann conditions

- $|\partial\Omega_\mathrm{D}|, |\partial\Omega_\mathrm{N}| > 0$, rearrange rows of A, entries of x and b:

$$A_{ik} = \begin{cases} G_{ik} \\ -\hat{F}_{ik} \end{cases} \qquad x_k = \begin{cases} \bar{q}_k & \partial\Omega_k \in \partial\Omega_\mathrm{D} \\ \bar{\phi}_k & \partial\Omega_k \in \partial\Omega_\mathrm{N} \end{cases}$$

$$b_i = G_{ij} q_j + \sum_{\substack{k \\ \partial\Omega_k \in \partial\Omega_\mathrm{D}}} \hat{F}_{ik} \bar{\phi}_k - \sum_{\substack{k \\ \partial\Omega_k \in \partial\Omega_\mathrm{N}}} G_{ik} \bar{q}_k$$

Periodic boundary conditions

- $\partial\Omega_\mathrm{P}$, pairs of walls discretised into mirroring $\partial\Omega_k \leftrightarrow \partial\Omega_{k'}$, $\bar{\phi}_k = \bar{\phi}_{k'}$, $\bar{q}_k = \bar{q}_{k'}$:

$$\begin{aligned} A_{ik} &= G_{ik} + G_{ik'} & \qquad x_k &= \bar{q}_k \\ A_{ik'} &= -\hat{F}_{ik} - \hat{F}_{ik'} & \qquad x_{k'} &= \bar{\phi}_{k'} \end{aligned}$$

One more boundary condition

Metal wall at floating potential

- $\partial\Omega_F$ made of metal
- $\bar{\phi} = \bar{\phi}_F =$ const but unknown on $\partial\Omega_F$
- $\bar{q}$ unknown, but total charge $Q_F = \int_{\partial\Omega_F} \bar{q}(x)\,dS$ known
- extend system:

$$
\begin{aligned}
A_{ik} &= G_{ik} & x_k &= \bar{q}_k & \partial\Omega_k \in \partial\Omega_F \\
A_{i(N+1)} &= -\sum_{\substack{k \\ \partial\Omega_k \in \partial\Omega_F}} F_{ik} & x_{N+1} &= \bar{\phi}_F & \\
A_{(N+1)k} &= \begin{cases} |\partial\Omega_k| & \partial\Omega_k \in \partial\Omega_F \\ 0 & \end{cases} & b_{N+1} &= Q_F &
\end{aligned}
$$